	Type	Hits	Search Text	Ref #	DBs
1	BRS	16860	object with propagat$5	S61	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
2	BRS	12	S52 and S61	S62	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
3	BRS	20	S55 and propagat$5	S56	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
4	BRS	25	S54 and construct$5	S55	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
5	BRS	20706	virtual$2 with direct$2	S53	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
6	BRS	25	S52 and S53	S54	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
7	BRS	369	"call graph"	S52	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
8	BRS	23	(S48 or S49) and virtual and direct	S50	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
9	BRS	5	S50 and propagat$3	S51	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
10	BRS	23	S45 near8 type	S47	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
11	IS&R	54	(717/133).CCLS.	S48	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
12	IS&R	62	(717/157).CCLS.	S49	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
13	BRS	1434	(recipro$5 or mutual$3) near4 propagat$5	S45	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
14	BRS	0	S38 and S45	S46	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
15	BRS	4	S38 and S43	S44	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
16	BRS	31	S41 and propagat$3	S42	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
17	BRS	7691	type near4 propagat$3	S43	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
18	BRS	42	("call graph" with construct$5) and object	S2	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
19	BRS	65	("call graph" with construct$5) and object	S41	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
20	BRS	1124	virtual$3 with direct$3 with call$3	S38	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
21	BRS	179	S38 and propagat$3	S39	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
22	BRS	44	S39 and graph	S40	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
23	BRS	49	"call graph" with construct$5	S1	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
24	IS&R	356	(717/116).CCLS.	S34	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB

	Time Stamp	Comments	Error Definition	Errors
1	2005/06/13 15:53			
2	2005/06/13 15:53			
3	2005/06/13 15:49			
4	2005/06/13 14:57			
5	2005/06/13 14:53			
6	2005/06/13 14:53			
7	2005/06/13 14:52			
8	2005/05/02 10:15			
9	2005/05/02 10:15			
10	2005/05/02 10:13			
11	2005/05/02 10:13			
12	2005/05/02 10:13			
13	2005/05/01 21:00			
14	2005/05/01 21:00			
15	2005/05/01 20:59			
16	2005/05/01 20:34			
17	2005/05/01 20:34			
18	2005/05/01 19:51			
19	2005/05/01 19:51			
20	2005/05/01 19:46			
21	2005/05/01 19:46			
22	2005/05/01 19:46			
23	2005/05/01 19:45			
24	2005/01/08 20:01			

25	IS&R	276	(717/131).CCLS.	S35	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB

25	2005/01/08 20:01			

	Type	Hits	Search Text	Ref #	DBs
26	IS&R	1244	(711/202).CCLS.	S36	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
27	IS&R	1897	(707/2).CCLS.	S37	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
28	IS&R	269	(717/130).CCLS.	S33	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
29	IS&R	57	(717/157).CCLS.	S31	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
30	IS&R	53	(717/132).CCLS.	S32	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB
31	BRS	0	methd same body same program	S26	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
32	BRS	3367	method same body same program	S27	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
33	BRS	9453	direct same virtual	S28	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
34	BRS	51	S27 and S28	S29	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
35	BRS	9	S29 and call and graph	S30	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
36	BRS	19	S21 and "run-time"	S25	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
37	BRS	4	"virtually called"	S23	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
38	BRS	1	S21 and S22	S24	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
39	BRS	55	"allocation site"	S15	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
40	BRS	63	"allocation site"	S21	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB
41	BRS	479	"directly called"	S22	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB

	Time Stamp	Comments	Error Definition	Errors
26	2005/01/08 20:01			
27	2005/01/08 20:01			
28	2005/01/08 19:58			
29	2005/01/08 19:57			
30	2005/01/08 19:57			
31	2005/01/08 17:34			
32	2005/01/08 17:34			
33	2005/01/08 17:34			
34	2005/01/08 17:34			
35	2005/01/08 17:34			
36	2005/01/08 17:33			
37	2005/01/08 17:15			
38	2005/01/08 17:15			
39	2005/01/08 17:14			
40	2005/01/08 17:14			
41	2005/01/08 17:14			

P◈RTAL
USPTO

THE ACM DIGITAL LIBRARY

⚑ Feedback Report a problem Satisfaction survey

Terms used **propagation based**   Found **114** of **156,259**

Sort results by | relevance ▽ | ✎ Save results to a Binder   Try an Advanced Search
Display results | expanded form ▽ | ？ Search Tips   Try this search in The ACM Guide
                                                   ☐ Open results in a new window

Results 1 - 20 of 114   Result page: **1** 2 3 4 5 6   next

Relevance scale ☐ ☐ ☐ ■ ■

1 Scalable propagation-based call graph construction algorithms   ■
Frank Tip, Jens Palsberg
October 2000 **ACM SIGPLAN Notices , Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 35 Issue 10

Full text available: 📄 pdf(677.36 KB)   Additional Information: full citation, abstract, references, citings, index terms

Propagation-based call graph construction algorithms have been studied intensively in the 1990s, and differ primarily in the number of sets that are used to approximate run-time values of expressions. In practice, algorithms such as RTA that use a single set for the whole program scale well. The scalability of algorithms such as 0-CFA that use one set per expression remains doubtful.In this paper, we investigate the design space between RTA and 0-CFA. We have implemented various novel algorithms ...

2 On the interconnection of causal memory systems   ■
Antonio Fernández, Ernesto Jiménez, Vicent Cholvi
July 2000 **Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing**

Full text available: 📄 pdf(792.05 KB)   Additional Information: full citation, abstract, references, index terms

A large amount of work has been invested in devising algorithms to implement distributed shared memory (DSM) systems under different consistency models. However, to our knowledge, the possibility of interconnecting DSM systems with simple protocols and the consistency of the resulting system has never been studied. With this paper, we start a series of works on the properties of the interconnection of DSM systems, which tries to fill this void. In this paper, we look at the inter ...

3 Cache investment: integrating query optimization and distributed data placement   ☐
Donald Kossmann, Michael J. Franklin, Gerhard Drasch, Wig Ag
December 2000 **ACM Transactions on Database Systems (TODS)**, Volume 25 Issue 4

Full text available: 📄 pdf(210.67 KB)   Additional Information: full citation, abstract, references, citings, index terms

Emerging distributed query-processing systems support flexible execution strategies in which each query can be run using a combination of data shipping and query shipping. As in any distributed environment, these systems can obtain tremendous performance and availability benefits by employing dynamic data caching. When flexible execution and dynamic caching are combined, however, a circular dependency arises: Caching occurs as a by-product of query operator placement, but query operator pl ...

Keywords: cache investment, caching, client-server database systems, data shipping, dynamic data placement, query optimization, query shipping

4 When do bounds and domain propagation lead to the same search space

Christian Schulte, Peter J. Stuckey

September 2001 **Proceedings of the 3rd ACM SIGPLAN international conference on Principles and practice of declarative programming**

Full text available: pdf(295.88 KB)   Additional Information: full citation, abstract, references, index terms

This paper explores the question of when two propagation-based constraint systems have the same behaviour, in terms of search space. We categorise the behaviour of domain and bounds propagators for primitive constraints, and provide theorems that allow us to determine propagation behaviours for conjunctions of constraints. We then show how we can use this to analyse CLP(FD) programs to determine when we can safely replace domain propagators by more efficient bounds propagators without increasing ...

Keywords: abstract interpretation, bounds propagation, constraint (logic) programming, domain propagation, finite domain constraints, program analysis

5 Access rights analysis for Java

Larry Koved, Marco Pistoia, Aaron Kershenbaum

November 2002 **ACM SIGPLAN Notices , Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 37 Issue 11

Full text available: pdf(360.93 KB)   Additional Information: full citation, abstract, references, citings, index terms

Java 2 has a security architecture that protects systems from unauthorized access by mobile or statically configured code. The problem is in manually determining the set of security access rights required to execute a library or application. The commonly used strategy is to execute the code, note authorization failures, allocate additional access rights, and test again. This process iterates until the code successfully runs for the test cases in hand. Test cases usually do not cover all paths th ...

Keywords: Java security, access rights, call graph, data flow analysis, invocation graph, security

6 A framework for call graph construction algorithms

David Grove, Craig Chambers

November 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 23 Issue 6

Full text available: pdf(1.36 MB)   Additional Information: full citation, abstract, references, citings, index terms

A large number of call graph construction algorithms for object-oriented and functional languages have been proposed, each embodying different tradeoffs between analysis cost and call graph precision. In this article we present a unifying framework for understanding call graph construction algorithms and an empirical comparison of a representative set of algorithms. We first present a general parameterized algorithm that encompasses many well-known and novel call graph construction algorithms. W ...

Keywords: Call graph construction, control flow analysis, interprocedural analysis

7 Materialized view selection and maintenance using multi-query optimization

Hoshi Mistry, Prasan Roy, S. Sudarshan, Krithi Ramamritham

May 2001 **ACM SIGMOD Record , Proceedings of the 2001 ACM SIGMOD international conference on Management of data**, Volume 30 Issue 2

Full text available: pdf(199.46 KB)   Additional Information: full citation, abstract, references, citings, index terms

Materialized views have been found to be very effective at speeding up queries, and are increasingly being supported by commercial databases and data warehouse systems. However, whereas the amount of data entering a warehouse and the number of materialized views are rapidly increasing, the time window available for maintaining materialized views is shrinking. These trends necessitate efficient techniques for the maintenance of materialized views.

In this paper, we show how to find an ...

8 **An empirical study of non-binary genetic algorithm-based neural approaches for classification**
Parag C. Pendharkar, James A. Rodger
January 1999 **Proceeding of the 20th international conference on Information Systems**

Full text available: pdf(192.45 KB)   Additional Information: full citation, references, citings, index terms

9 **XML security: Concept-level access control for the Semantic Web**
Li Qin, Vijayalakshmi Atluri
October 2003 **Proceedings of the 2003 ACM workshop on XML security**

Full text available: pdf(320.46 KB)   Additional Information: full citation, abstract, references, index terms

Recently, the notion of the Semantic Web has been introduced to define a machine-interpretable web targeted for automation, integration and reuse of data across different applications. Under the Semantic Web, web pages are annotated by concepts that are formally defined in ontologies along with the relationships among them. As information pertaining to different concepts has varying access control requirements, in this paper, we propose an access control model for the semantic web that is capabl ...

Keywords: Semantic Web, access control, concept, ontology, propagation

10 **Class analyses as abstract interpretations of trace semantics**
Fausto Spoto, Thomas Jensen
September 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,
Volume 25 Issue 5
Full text available: pdf(756.68 KB)   Additional Information: full citation, abstract, references, index terms

We use abstract interpretation to abstract a compositional trace semantics for a simple imperative object-oriented language into its projection over a set of program points called *watchpoints*. We say that the resulting *watchpoint semantics* is *focused* on the watchpoints. Every abstraction of the computational domain of this semantics induces an abstract, still compositional, and focused watchpoint semantics. This establishes a basis for developing static analyses obtaining in ...

Keywords: Abstract interpretation, class analysis, denotational semantics

11 **Formation and simulation: Worm propagation modeling and analysis under dynamic quarantine defense**
Cliff Changchun Zou, Weibo Gong, Don Towsley
October 2003 **Proceedings of the 2003 ACM workshop on Rapid Malcode**

Full text available: pdf(265.77 KB)   Additional Information: full citation, abstract, references, citings, index terms

Due to the fast spreading nature and great damage of Internet worms, it is necessary to implement automatic mitigation, such as dynamic quarantine, on computer networks. Enlightened by the methods used in epidemic disease control in the real world, we present a dynamic quarantine method based on the principle "assume guilty before proven innocent" --- we quarantine a host whenever its behavior looks suspicious by blocking traffic on its

anomaly port. Then we will release the quarantine after a s ...

Keywords: dynamic quarantine, epidemic model, worm propagation

12 Software techniques for program compaction: Extracting library-based Java applications
Frank Tip, Peter F. Sweeney, Chris Laffra
August 2003 **Communications of the ACM**, Volume 46 Issue 8

Full text available: pdf(235.95 KB)   Additional Information: full citation, abstract, references, index terms,
html(28.30 KB)   review

Reducing the size of Java applications by creating an application extractor.

13 Noise propagation and failure criteria for VLSI designs
V. Zolotov, D. Blaauw, S. Sirichotiyakul, M. Becer, C. Oh, R. Panda, A. Grinshpon, R. Levy
November 2002 **Proceedings of the 2002 IEEE/ACM international conference on Computer-aided design**

Full text available: pdf(237.68 KB)   Additional Information: full citation, abstract, references, citings, index terms

Noise analysis has become a critical concern in advanced chip designs. Traditional methods suffer from two common issues. First, noise that is propagated through the driver of a net is combined with noise injected by capacitively coupled aggressor nets using linear summation. Since this ignores the non-linear behavior of the driver gate the noise that develops on a net can be significantly underestimated. We therefore propose a new linear model that accurately combines propagated and injected no ...

14 Technical papers: testing II: Fragment class analysis for testing of polymorphism in Java software
Atanas Rountev, Ana Milanova, Barbara G. Ryder
May 2003 **Proceedings of the 25th International Conference on Software Engineering**

Full text available: pdf(1.13 MB)   Additional Information: full citation, abstract, references, citings, index terms
Publisher Site

Adequate testing of polymorphism in object-oriented software requires coverage of all possible bindings of receiver classes and target methods at call sites. Tools that measure this coverage need to use *class analysis* to compute the coverage requirements. However, traditional whole-program class analysis cannot be used when testing partial programs. To solve this problem, we present a general approach for adapting whole-program class analyses to operate on program fragments. Furthermore, ...

15 Practical extraction techniques for Java
Frank Tip, Peter F. Sweeney, Chris Laffra, Aldo Eisma, David Streeter
November 2002 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 24 Issue 6

Full text available: pdf(1.01 MB)   Additional Information: full citation, abstract, references, citings, index terms, review

Reducing application size is important for software that is distributed via the internet, in order to keep download times manageable, and in the domain of embedded systems, where applications are often stored in (Read-Only or Flash) memory. This paper explores extraction techniques such as the removal of unreachable methods and redundant fields, inlining of method calls, and transformation of the class hierarchy for reducing application size. We implemented a number of extraction techniques in < ...

Keywords: Application extraction, call graph construction, class hierarchy transformation, packaging, whole-program analysis

16 Papers: novel input, output, and computation: Dynamic approximation of complex graphical constraints by linear constraints

Nathan Hurst, Kim Marriott, Peter Moulder

October 2002 **Proceedings of the 15th annual ACM symposium on User interface software and technology**

Full text available: 📄 pdf(397.65 KB) Additional Information: full citation, abstract, references, citings, index terms

Current constraint solving techniques for interactive graphical applications cannot satisfactorily handle constraints such as non-overlap, or containment within non-convex shapes or shapes with smooth edges. We present a generic new technique for efficiently handling such kinds of constraints based on trust regions and linear arithmetic constraint solving. Our approach is to model these more complex constraints by a dynamically changing conjunction of linear constraints. At each stage, these giv ...

Keywords: constraint-solving, containment, direct manipulation, linearization of constraints, non-overlap, trust regions

17 A conservative algorithm for computing the flow of permissions in Java programs

Gleb Naumovich

July 2002 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2002 ACM SIGSOFT international symposium on Software testing and analysis**, Volume 27 Issue 4

Full text available: 📄 pdf(540.08 KB) Additional Information: full citation, abstract, references, citings

Open distributed systems are becoming increasingly popular. Such systems include components that may be obtained from a number of different sources. For example, Java allows run-time loading of software components residing on remote machines. One unfortunate side-effect of this openness is the possibility that "hostile" software components may compromise the security of both the program and the system on which it runs. Java offers a built-in security mechanism, using which programmers can give p ...

Keywords: data flow analysis, java, security, static analysis, verification

18 Dynamic optimistic interprocedural analysis: a framework and an application

Igor Pechtchanski, Vivek Sarkar

October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11

Full text available: 📄 pdf(1.78 MB) Additional Information: full citation, abstract, references, citings, index terms, review

In this paper, we address the problem of *dynamic optimistic interprocedural analysis*. Our goal is to build on past work on *static interprocedural analysis* and *dynamic optimization* by combining their advantages. We present a framework for performing dynamic optimistic interprocedural analysis. the framework is designed to be used in the context of dynamic class loading and dynamic compilation, and includes mechanisms for event notification (on class loading and method compila ...

19 Staged compilation

Craig Chambers

January 2002 **ACM SIGPLAN Notices , Proceedings of the 2002 ACM SIGPLAN workshop on Partial evaluation and semantics-based program manipulation**, Volume 37 Issue 3

Full text available: 📄 pdf(80.45 KB) Additional Information: full citation, abstract, references, citings

Traditional compilers compile and optimize files separately, making worst-case assumptions about the program context in which a file is to be linked. More aggressive compilation architectures perform cross-file interprocedural or whole-program analyses, potentially producing much faster programs but substantially increasing the cost of compilation. Even

more radical are systems that perform all compilation and optimization at run-time: such systems can optimize programs based on run-time program ...

20 <u>On the syllogistic structure of object-oriented programming</u>

Derek Rayside, Kostas Kontogiannis

July 2001 **Proceedings of the 23rd International Conference on Software Engineering**

Full text available: <u>pdf(138.07 KB)</u>   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

<u>Publisher Site</u>

Recent works by Sowa and by Rayside & Campbell demonstrate that there is a strong connection between object-oriented programming and the logical formalism of the syllogism, first set down by Aristotle in the Prior Analytics. *In this paper, we develop an understanding of polymorphic method invocations in terms of the syllogism, and apply this understanding to the design of a novel editor for object-oriented programs. This editor is able to display a polymorphic call graph, which ...*

Results 1 - 20 of 114   Result page: **1** <u>2</u> <u>3</u> <u>4</u> <u>5</u> <u>6</u>   <u>next</u>

Useful downloads: <u>Adobe Acrobat</u>   <u>QuickTime</u>   <u>Windows Media Player</u>   <u>Real Player</u>

P RTAL

USPTO

Search: ◉ The ACM Digital Library   ○ The Guide

`"propagation based" +"call graph"`   **SEARCH**

THE ACM DIGITAL LIBRARY

ι⃦ <u>Feedback</u> <u>Report a problem</u> <u>Satisfaction survey</u>

Terms used **propagation based** **call graph**   Found **27** of **114 searched out of 114.**

Sort results by   relevance ▽   ♦ <u>Save results to a Binder</u>

Display results   expanded form ▽   ? <u>Search Tips</u>
                                            ☐ Open results in a new window

Try an <u>Advanced Search</u>
Try this search in <u>The ACM Guide</u>

Results 1 - 20 of 27   Result page: **1**   <u>2</u>   <u>next</u>

Relevance scale ☐ ▭ ◼ ◼ ◼

1 <u>A framework for call graph construction algorithms</u>   ◼
David Grove, Craig Chambers
November 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS),**
                Volume 23 Issue 6

Full text available: 📄<u>pdf(1.36 MB)</u>   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

A large number of call graph construction algorithms for object-oriented and functional languages have been proposed, each embodying different tradeoffs between analysis cost and call graph precision. In this article we present a unifying framework for understanding call graph construction algorithms and an empirical comparison of a representative set of algorithms. We first present a general parameterized algorithm that encompasses many well-known and novel call graph construction algorithms. W ...

Keywords: Call graph construction, control flow analysis, interprocedural analysis

2 <u>Scalable propagation-based call graph construction algorithms</u>   ◼
Frank Tip, Jens Palsberg
October 2000 **ACM SIGPLAN Notices , Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications,** Volume 35 Issue 10

Full text available: 📄<u>pdf(677.36 KB)</u>   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

Propagation-based call graph construction algorithms have been studied intensively in the 1990s, and differ primarily in the number of sets that are used to approximate run-time values of expressions. In practice, algorithms such as RTA that use a single set for the whole program scale well. The scalability of algorithms such as 0-CFA that use one set per expression remains doubtful.In this paper, we investigate the design space between RTA and 0-CFA. We have implemented various novel algorithms ...

3 <u>Practical extraction techniques for Java</u>   ◼
Frank Tip, Peter F. Sweeney, Chris Laffra, Aldo Eisma, David Streeter
November 2002 **ACM Transactions on Programming Languages and Systems (TOPLAS),**
                Volume 24 Issue 6

Full text available: 📄<u>pdf(1.01 MB)</u>   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>, <u>review</u>

Reducing application size is important for software that is distributed via the internet, in order to keep download times manageable, and in the domain of embedded systems, where applications are often stored in (Read-Only or Flash) memory. This paper explores extraction techniques such as the removal of unreachable methods and redundant fields,

inlining of method calls, and transformation of the class hierarchy for reducing application size. We implemented a number of extraction techniques in < ...

Keywords: Application extraction, call graph construction, class hierarchy transformation, packaging, whole-program analysis

4 Practical virtual method call resolution for Java

Vijay Sundaresan, Laurie Hendren, Chrislain Razafimahefa, Raja Vallée-Rai, Patrick Lam, Etienne Gagnon, Charles Godin

October 2000 **ACM SIGPLAN Notices , Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 35 Issue 10

Full text available: 🔁 pdf(323.98 KB)   Additional Information: full citation, abstract, references, citings, index terms

This paper addresses the problem of resolving virtual method and interface calls in Java bytecode. The main focus is on a new practical technique that can be used to analyze large applications. Our fundamental design goal was to develop a technique that can be solved with only one iteration, and thus scales linearly with the size of the program, while at the same time providing more accurate results than two popular existing linear techniques, *class hierarchy analysis* and *rapid type an* ...

5 Access rights analysis for Java

Larry Koved, Marco Pistoia, Aaron Kershenbaum

November 2002 **ACM SIGPLAN Notices , Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 37 Issue 11

Full text available: 🔁 pdf(360.93 KB)   Additional Information: full citation, abstract, references, citings, index terms

Java 2 has a security architecture that protects systems from unauthorized access by mobile or statically configured code. The problem is in manually determining the set of security access rights required to execute a library or application. The commonly used strategy is to execute the code, note authorization failures, allocate additional access rights, and test again. This process iterates until the code successfully runs for the test cases in hand. Test cases usually do not cover all paths th ...

Keywords: Java security, access rights, call graph, data flow analysis, invocation graph, security

6 A conservative algorithm for computing the flow of permissions in Java programs

Gleb Naumovich

July 2002 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2002 ACM SIGSOFT international symposium on Software testing and analysis**, Volume 27 Issue 4

Full text available: 🔁 pdf(540.08 KB)   Additional Information: full citation, abstract, references, citings

Open distributed systems are becoming increasingly popular. Such systems include components that may be obtained from a number of different sources. For example, Java allows run-time loading of software components residing on remote machines. One unfortunate side-effect of this openness is the possibility that "hostile" software components may compromise the security of both the program and the system on which it runs. Java offers a built-in security mechanism, using which programmers can give p ...

Keywords: data flow analysis, java, security, static analysis, verification

7 Points-to analysis for Java using annotated constraints

Atanas Rountev, Ana Milanova, Barbara G. Ryder

October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11

Full text available: pdf(263.51 KB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

The goal of point-to analysis for Java is to determine the set of objects pointed by a reference variable or a reference object field. This information has a wide variety of client applications in optimizing compilers and software engineering tools. In this paper we present a point-to analysis for Java based on Andersen's point-to analysis for C [5]. We implement the analysis by using a constraint-based approach which employs *annotated inclusion constraints.* Constraint annotations allow u ...

8 <u>Parameterized object sensitivity for points-to analysis for Java</u>

Ana Milanova, Atanas Rountev, Barbara G. Ryder

January 2005 **ACM Transactions on Software Engineering and Methodology (TOSEM)**, Volume 14 Issue 1

Full text available: pdf(413.28 KB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>index terms</u>

The goal of *points-to analysis* for Java is to determine the set of objects pointed to by a reference variable or a reference object field. We present *object sensitivity*, a new form of context sensitivity for flow-insensitive points-to analysis for Java. The key idea of our approach is to analyze a method separately for each of the object names that represent run-time objects on which this method may be invoked. To ensure flexibility and practicality, we propose a parameterization f ...

Keywords: Static analysis, class analysis, context sensitivity, def-use analysis, points-to analysis, side-effect analysis

9 <u>On the syllogistic structure of object-oriented programming</u>

Derek Rayside, Kostas Kontogiannis

July 2001 **Proceedings of the 23rd International Conference on Software Engineering**

Full text available: pdf(138.07 KB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>
           Publisher Site

Recent works by Sowa and by Rayside & Campbell demonstrate that there is a strong connection between object-oriented programming and the logical formalism of the syllogism, first set down by Aristotle in the Prior Analytics. In this paper, we develop an understanding of polymorphic method invocations in terms of the syllogism, and apply this understanding to the design of a novel editor for object-oriented programs. This editor is able to display a polymorphic call graph, which ...

10 <u>Extracting library-based object-oriented applications</u>

Peter F. Sweeney, Frank Tip

November 2000 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 8th ACM SIGSOFT international symposium on Foundations of software engineering: twenty-first century applications**, Volume 25 Issue 6

Full text available: pdf(1.06 MB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

In an increasingly popular model of software distribution, software is developed in one computing environment and deployed in other environments by transfer over the internet. Extraction tools perform a static whole-program analysis to determine unused functionality in applications in order to reduce the time required to download applications. We have identified a number of scenarios where extraction tools require information beyond what can be inferred through static analysis: software distr ...

11 <u>Parameterized object sensitivity for points-to and side-effect analyses for Java</u>

Ana Milanova, Atanas Rountev, Barbara G. Ryder

July 2002 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2002 ACM**

SIGSOFT international symposium on Software testing and analysis, Volume 27 Issue 4

Full text available: 📄 pdf(207.18 KB)   Additional Information: full citation, abstract, references, citings

The goal of *points-to analysis* for Java is to determine the set of objects pointed to by a reference variable or a reference objet field. Improving the precision of practical points-to analysis is important because points-to information has a wide variety of client applications in optimizing compilers and software engineering tools. In this paper we present *object sensitivity,* a new form of context sensitivity for flow-insensitive points-to analysis for Java. The key idea of our ap ...

12 Software techniques for program compaction: Extracting library-based Java applications

Frank Tip, Peter F. Sweeney, Chris Laffra
August 2003 **Communications of the ACM**, Volume 46 Issue 8

Full text available: 📄 pdf(235.95 KB)   Additional Information: full citation, abstract, references, index terms,
                  📄 html(28.30 KB)   review

Reducing the size of Java applications by creating an application extractor.

13 Dynamic optimistic interprocedural analysis: a framework and an application

Igor Pechtchanski, Vivek Sarkar
October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11

Full text available: 📄 pdf(1.78 MB)   Additional Information: full citation, abstract, references, citings, index
                                                        terms, review

In this paper, we address the problem of *dynamic optimistic interprocedural analysis*. Our goal is to build on past work on *static interprocedural analysis* and *dynamic optimization* by combining their advantages. We present a framework for performing dynamic optimistic interprocedural analysis. the framework is designed to be used in the context of dynamic class loading and dynamic compilation, and includes mechanisms for event notification (on class loading and method compila ...

14 Evaluation: An improved slicer for Java

Christian Hammer, Gregor Snelting
June 2004 **Proceedings of the ACM-SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering**

Full text available: 📄 pdf(180.49 KB)   Additional Information: full citation, abstract, references, index terms

We present an improved slicing algorithm for Java. The best algorithm known so far, first presented in [11], is not always precise if nested objects are used as actual parameters. The new algorithm presented in this paper always generates correct and precise slices, but is more expensive in general.We describe the algorithms and their treatment of objects as parameters. In particular, we present a new, safe criterion for termination of unfolding nested parameter objects. We then compare the two ...

Keywords: Java, object trees, static program slicing

15 Change impact analysis for object-oriented programs

Barbara G. Ryder, Frank Tip
June 2001 **Proceedings of the 2001 ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering**

Full text available: 📄 pdf(199.26 KB)   Additional Information: full citation, abstract, references, citings, index
                                                        terms

Small changes can have major and nonlocal effects in object-oriented languages, due to the use of subtyping and dynamic dispatch. This complicates life for maintenance programmers, who need to fix bugs or add enhancements to systems originally written by others. Change

impact analysis provides feedback on the semantic impact of a set of program changes. This analysis can be used to determine the regression test drivers that are affected by a set of changes. Moreover, if a t ...

16 Technical papers: testing II: Fragment class analysis for testing of polymorphism in Java software

Atanas Rountev, Ana Milanova, Barbara G. Ryder
May 2003 **Proceedings of the 25th International Conference on Software Engineering**

Full text available: pdf(1.13 MB) Publisher Site   Additional Information: full citation, abstract, references, citings, index terms

Adequate testing of polymorphism in object-oriented software requires coverage of all possible bindings of receiver classes and target methods at call sites. Tools that measure this coverage need to use *class analysis* to compute the coverage requirements. However, traditional whole-program class analysis cannot be used when testing partial programs. To solve this problem, we present a general approach for adapting whole-program class analyses to operate on program fragments. Furthermore, ...

17 Staged compilation

Craig Chambers
January 2002 **ACM SIGPLAN Notices , Proceedings of the 2002 ACM SIGPLAN workshop on Partial evaluation and semantics-based program manipulation**, Volume 37 Issue 3

Full text available: pdf(80.45 KB)   Additional Information: full citation, abstract, references, citings

Traditional compilers compile and optimize files separately, making worst-case assumptions about the program context in which a file is to be linked. More aggressive compilation architectures perform cross-file interprocedural or whole-program analyses, potentially producing much faster programs but substantially increasing the cost of compilation. Even more radical are systems that perform all compilation and optimization at run-time: such systems can optimize programs based on run-time program ...

18 Type-based analysis and applications

Jens Palsberg
June 2001 **Proceedings of the 2001 ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering**

Full text available: pdf(184.50 KB)   Additional Information: full citation, abstract, references, citings, index terms

Type-based analysis is an approach to static analysis of programs that has been studied for more than a decade. A type-based analysis assumes that the program type checks, and the analysis takes advantage of that. This paper examines the state of the art of type-based analysis, and it surveys some of the many software tools that use type-based analysis. Most of the surveyed tools use types as discriminators, while most of the theoretical studies use type and effect systems. We conclude that ...

19 Class analyses as abstract interpretations of trace semantics

Fausto Spoto, Thomas Jensen
September 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 25 Issue 5

Full text available: pdf(756.68 KB)   Additional Information: full citation, abstract, references, index terms

We use abstract interpretation to abstract a compositional trace semantics for a simple imperative object-oriented language into its projection over a set of program points called *watchpoints*. We say that the resulting *watchpoint semantics* is *focused* on the watchpoints. Every abstraction of the computational domain of this semantics induces an abstract, still compositional, and focused watchpoint semantics. This establishes a basis for developing static analyses obtaining in ...

Keywords: Abstract interpretation, class analysis, denotational semantics

20 <u>Sifting out the mud: low level C++ code reuse</u>   ■
Bjorn De Sutter, Bruno De Bus, Koen De Bosschere
November 2002 **ACM SIGPLAN Notices , Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 37 Issue 11

Full text available: <u>pdf(1.35 MB)</u>   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

More and more computers are being incorporated in devices where the available amount of memory is limited. This contrasts with the increasing need for additional functionality and the need for rapid application development. While object-oriented programming languages, providing mechanisms such as inheritance and templates, allow fast development of complex applications, they have a detrimental effect on program size. This paper introduces new techniques to reuse the code of whole procedures at t ...

Keywords: code compaction, code size reduction
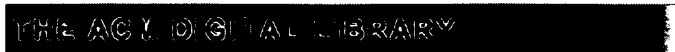
Results 1 - 20 of 27   Result page: **1**   <u>2</u>   <u>next</u>

Useful downloads: <u>Adobe Acrobat</u>   <u>QuickTime</u>   <u>Windows Media Player</u>   <u>Real Player</u>

PORTAL

USPTO

Search: ◉ The ACM Digital Library   ○ The Guide

"propagation based" +"call graph"   **SEARCH**

THE ACM DIGITAL LIBRARY

❡ Feedback Report a problem Satisfaction survey

Terms used **propagation based call graph**   Found **27** of **114**

Sort results by | relevance ▽ | **❧** Save results to a Binder   Try an Advanced Search
Display results | expanded form ▽ | **?** Search Tips   Try this search in The ACM Guide
☐ Open results in a new window

Results 21 - 27 of 27   Result page: previous **1** **2**

Relevance scale ☐ ▭ ▪ ◼ ■

21 Converting java programs to use generic libraries   ■

Alan Donovan, Adam Kiežun, Matthew S. Tschantz, Michael D. Ernst
October 2004 **ACM SIGPLAN Notices , Proceedings of the 19th annual ACM SIGPLAN Conference on Object-oriented programming, systems, languages, and applications**, Volume 39 Issue 10

Full text available: 📄 pdf(1.18 MB)   Additional Information: full citation, abstract, references, index terms

Java 1.5 will include a type system (called JSR-14) that supports <i>parametric polymorphism</i>, or <i>generic</i> classes. This will bring many benefits to Java programmers, not least because current Java practice makes heavy use of logically-generic classes, including container classes.

Translation of Java source code into semantically equivalent JSR-14 source code requires two steps: parameterization (adding type parameters to class definitions) and instantiation (a ...

Keywords: JSR-14, Java 1.5, Java 5, generic types, instantiation types, parameterized types, parametric polymorphism, raw types, type inference

22 Extending and evaluating flow-insenstitive and context-insensitive points-to analyses for Java   ■

Donglin Liang, Maikel Pennings, Mary Jean Harrold
June 2001 **Proceedings of the 2001 ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering**

Full text available: 📄 pdf(169.44 KB)   Additional Information: full citation, abstract, references, citings, index terms

This paper presents extensions to Steensgaard's and Andersen's algorithms to handle Java features. Without careful consideration, the handling of these features may affect the correctness, precision, and efficiency of these algorithms. The paper also presents the results of empirical studies. These studies compare the precision and efficiency of these two algorithms and evaluate the effectiveness of handling Java features using alternative approaches. The studies also evaluate the impact o ...

23 A study of devirtualization techniques for a Java Just-In-Time compiler   ■

Kazuaki Ishizaki, Motohiro Kawahito, Toshiaki Yasue, Hideaki Komatsu, Toshio Nakatani
October 2000 **ACM SIGPLAN Notices , Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 35 Issue 10

Full text available: 📄 pdf(225.89 KB)   Additional Information: full citation, abstract, references, citings, index terms

Many devirtualization techniques have been proposed to reduce the runtime overhead of dynamic method calls for various object-oriented languages, however, most of them are less effective or cannot be applied for Java in a straightforward manner. This is partly because Java is a statically-typed language and thus transforming a dynamic call to a static one does not make a tangible performance gain (owing to the low overhead of accessing the method table) unless it is inlined, and partly because t ...

24 Granularity of constraint-based analysis for Java

Byeong-Mo Chang, Jangwu Jo
September 2001 **Proceedings of the 3rd ACM SIGPLAN international conference on Principles and practice of declarative programming**
Full text available: pdf(244.23 KB)   Additional Information: full citation, abstract, references, index terms

This paper proposes a transformation-based approach to design constraint-based analyses for Java at a coarser granularity. In this approach, we design a less or equally precise but more efficient version of an original analysis by transforming the original construction rules into new ones. As applications of this rule transformation, we provide two instances of analysis design by rule-transformation. The first one designs a sparse version of class analysis for Java and the second one deals with ...

Keywords: constrain t-based analysis, construction rules, partition function, set constraints

25 Regression test selection for Java software

Mary Jean Harrold, James A. Jones, Tongyu Li, Donglin Liang, Ashish Gujarathi
October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11
Full text available: pdf(292.35 KB)   Additional Information: full citation, abstract, references, citings, index terms

Regression testing is applied to modified software to provide confidence that the changed parts behave as intended and that the unchanged parts have not been adversely affected by the modifications. To reduce the cost of regression testing, test cases are selected from the test suite that was used to test the original version of the software---this process is called regression test selection. A *safe* regression-test-selection algorithm selects every test case in the test suite that may rev ...

26 Direct update of data flow representations for a meaning-preserving program restructuring tool

William G. Griswold
December 1993 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 1st ACM SIGSOFT symposium on Foundations of software engineering**, Volume 18 Issue 5
Full text available: pdf(1.64 MB)   Additional Information: full citation, abstract, references, citings, index terms

Automated assistance for meaning-preserving global restructuring is an approach for helping software engineers improve the structure of programs, thus lowering the costs of maintenance. The consturction of a restructuing tool encounters many conflicting goals--- such as simplicity. extensibility, and good performance---that cannot be met without some compromise. In particular, the current technique for assisting restructuring uses a costly program representation---a Program Dependence Graph (PDG) ...

27 Fast interprocedural class analysis

Greg DeFouw, David Grove, Craig Chambers
January 1998 **Proceedings of the 25th ACM SIGPLAN-SIGACT symposium on Principles of programming languages**
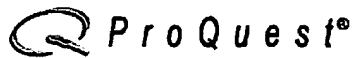Full text available: pdf(2.03 MB)   Additional Information: full citation, references, citings, index terms

Results 21 - 27 of 27   Result page: previous   1   **2**

Useful downloads: Adobe Acrobat   QuickTime   Windows Media Player   Real Player

ProQuest®

Basic Advanced Topics Publications My Research
0 marked Items

Databases selected: Multiple databases...   What's new

Results – powered by ProQuest® Smart Search

Suggested Topics About   < Previous | Next >   **Browse Suggested Publications** About   < Previous | Next >

Business cycles   The American Economic Review; Nashville
Business cycles AND Economic models

7 documents found for: *"propagation based"* Set up Alert About

(All sources) 🎓 Scholarly Journals   Trade Publications   Dissertations

☐ Mark all 📁 0 marked items: Email / Cite / Export   📄 Show only full text   Sort results by: Most recent first ▾

☐ 1. **Constraint-Propagation-Based Cutting Planes: An Application to the Resource-Constrained Project Scheduling Problem**
Sophie Demassey, Christian Artigues, Philippe Michelon. **INFORMS Journal on Computing.** Linthicum: Winter 2005. Vol. 17, Iss. 1; p. 52 (14 pages)

📖 Abstract

☐ 2. **Memetic Algorithm timetabling for non-commercial sport leagues**
J Schönberger, D C Mattfeld, H Kopfer. **European Journal of Operational Research.** Amsterdam: Feb 16, 2004. Vol. 153, Iss. 1; p. 102

📖 Abstract

☐ 3. **Connectionist models for learning, discovering and forecasting software effort: An empirical study**
Parag C Pendharkar, Girish H Subramanian. **The Journal of Computer Information Systems.** Stillwater: Fall 2002. Vol. 43, Iss. 1; p. 7 (8 pages)

📖 Text+Graphics   📄 Page Image - PDF   📖 Abstract

☐ 4. **Anisotropic conduction properties in canine atria analyzed by high-resolution optical mapping: preferential direction of conduction block changes from longitudinal to transverse with increasing age.**
Koura T, Hara M, Takeuchi S, Ota K, et al. **Circulation [NLM - MEDLINE].** Apr 30 2002. Vol. 105, Iss. 17; p. 2092

📖 Abstract

☐ 5. **A phase odyssey**
Keith A Nugent, David Paganin, Tim E Gureyev. **Physics Today.** New York: Aug 2001. Vol. 54, Iss. 8; p. 27

📖 Abstract

☐ 6. **A linear programming and constraint propagation-based lower bound for the RCPSP**
Peter Brucker, Sigrid Knust. **European Journal of Operational Research.** Amsterdam: Dec 1, 2000. Vol. 127, Iss. 2; p. 355

📖 Abstract

☐ 7. **Uncertainty Propagation and Speculation in Projective Forecasts of Environmental Change: A Lake-Eutrophication Example**
van Straten, Gerrit, Keesman, Karel J.. **Journal of Forecasting.** Chichester: Jan 1991. Vol. 10, Iss. 1,2; p. 163 (28 pages)

📄 Page Image - PDF   📖 Abstract

1-7 of 7

Want an alert for new results sent by email? [Set up Alert] About

Results per page: [30 ▼]

Did you find what you're looking for? If not, revise your search below or try these suggestions:

Suggested Topics About   < Previous | Next >   **Browse Suggested Publications** About   < Previous | Next >

Business cycles   The American Economic Review; Nashville

Business cycles AND Economic models

Basic Search

Tools: Search Tips   Browse Topics   3 Recent Searches

| "propagation based" | [Search] [Clear] |

Database: | Multiple databases...   ▼ | Select multiple databases

Date range: | All dates   ▼ |

Limit results to: ☐ Full text documents only 📄

                 ☐ Scholarly journals, including peer-reviewed 🎓 About

More Search Options